

Programación Orientada a Objetos

Proyecto Final

.

**PROFESOR:**

M.C. Alejandro Pasos Ruiz

.

**ELABORADO**

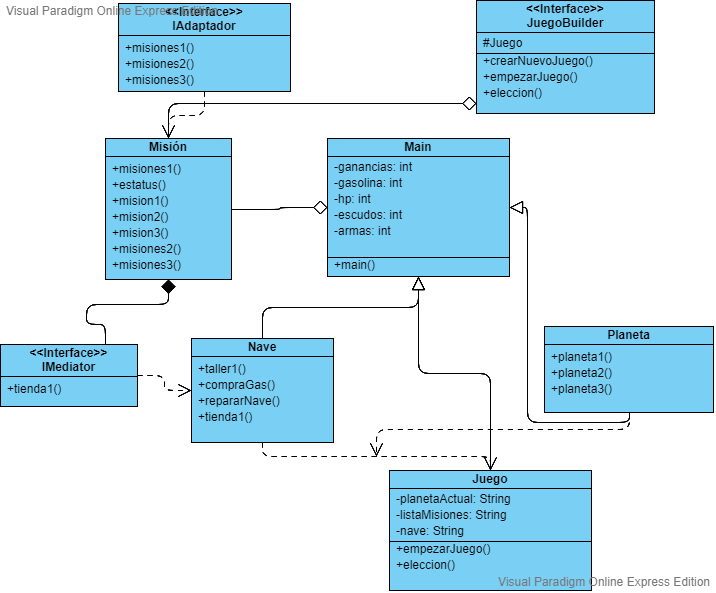
**:**

Baak Poot Nicolás Alejandro

Gongora Gamboa Diego Joel

Moreno Carvajal Pedro Abel

Juego en modo texto “Contrabandistas en el espacio”

Diagrama UML

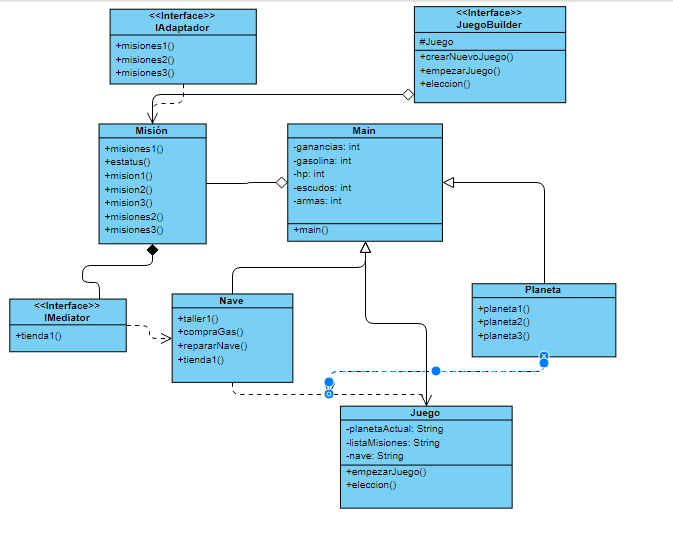
Para el diagrama UML del juego, incluimos los tres patrones de diseño que encontramos compatibles con nuestro programa, uno por cada tipo: para el patrón de tipo creacional elegimos el patrón Builder, para el de tipo estructural utilizamos el patrón Adapter; y para el patrón de comportamiento se utilizó Mediator.

* Patrón Creacional

Builder: Este patrón separa la construcción del objeto complejo juego (incluye todos sus atributos y métodos) así que podemos crear objetos con diferentes representaciones, para así crear un objeto, por ejemplo cuando iniciamos un nuevo juego en tiempo de ejecución, también sirve cuando cambiamos de planeta o nave.

Participantes para nuestro juego:

* Clase abstracta: JuegoBuilder
* Clase principal: Main
* Clase producto: Juego
* Clase director: Misión
* Concrete builders: Nave, Planeta

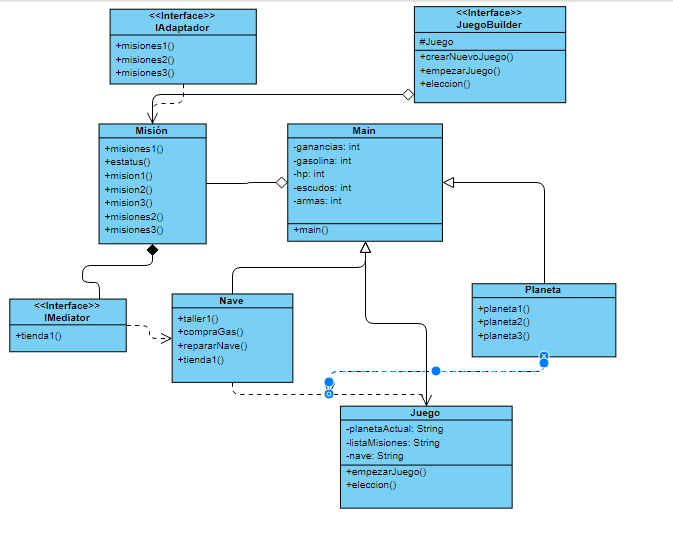


* Patrón Estructural

Adapter: Utilizamos este patrón para permitir que puedan trabajar las clases Juego y Misión con sus interfaces incompatibles. Así, un objeto adaptador o intermediario reenvía al otro objeto los datos que recibe, permitiendo entonces la interacción.

Participantes para nuestro juego:

* Clase Client: Main
* Interfaz: IAdaptador
* Clase adaptable: Juego



* Patrón de Comportamiento

Mediator: Definimos este patrón para encapsular la comunicación entre objetos (Clases misión y Nave), así se evita que lo hagan directo entre sí. Básicamente utilizamos el patrón para definir un intermediario que se encarga de gestionar la comunicación entre los objetos.

Participantes para nuestro juego:

* Interfaz: IMediador
* Clase mediadora: Nave
* Clase colega: Misión
* Clase colega concreta: Planeta

